Appears in *Proceedings of the 25th Annual ACM Symposium on the Theory of Computing,* ACM (1993).

Efficient Probabilistically Checkable Proofs and Applications to Approximation

M. Bellare[*] S. Goldwasser[†] C. Lund[‡] A. Russell[§]

May 1993

Abstract

We construct multi-prover proof systems for NP which use only a *constant* number of provers to simultaneously achieve low error, low randomness and low answer size. As a consequence, we obtain asymptotic improvements to approximation hardness results for a wide range of optimization problems including minimum set cover, dominating set, maximum clique, chromatic number, and quartic programming; and constant factor improvements on the hardness results for MAXSNP problems.

In particular, we show that approximating minimum set cover within *any* constant is NP-complete; approximating minimum set cover within $\Theta(\log n)$ implies $\mathrm{NP} \subseteq \mathrm{DTIME}(n^{\log\log n})$; approximating the maximum of a quartic program within any constant is NP-hard; approximating maximum clique within $n^{1/30}$ implies $\mathrm{NP} \subseteq \mathrm{BPP}$; approximating chromatic number within $n^{1/146}$ implies $\mathrm{NP} \subseteq \mathrm{BPP}$; and approximating MAX-3SAT within 113/112 is NP-complete.

[*]Department of Computer Science & Engineering, Mail Code 0114, University of California at San Diego, 9500 Gilman Drive, La Jolla, CA 92093. E-mail: `mihir@cs.ucsd.edu`.

[†] MIT Laboratory for Computer Science, 545 Technology Square, Cambridge, MA 02139, USA. e-mail: `shafi@theory.lcs.mit.edu`. Partially supported by NSF FAW grant No. 9023312-CCR, DARPA grant No. N00014-92-J-1799, and grant No. 89-00312 from the United States - Israel Binational Science Foundation (BSF), Jerusalem, Israel.

[‡]AT&T Bell Laboratories, Room 2C324, 600 Mountain Avenue, P. O. Box 636, Murray Hill, NJ 07974-0636, USA. email: `lund@research.att.com`.

[§] MIT Laboratory for Computer Science, 545 Technology Square, Cambridge, MA 02139, USA. e-mail: `acr@theory.lcs.mit.edu`. Supported by a NSF Graduate Fellowship and by NSF grant 92-12184, AFOSR 89-0271, and DARPA N00014-92-J-1799.

1 Introduction

The last two years have witnessed major advances in classifying the complexity of approximating classical optimization problems [Co, FGLSS, AS, ALMSS, BR, FL, Be, Zu, LY1, LY2]. These works indicate that a problem P is hard to approximate by showing that the existence of a polynomial time algorithm that approximates problem P to within some factor Q would imply an unlikely conclusion like $\mathrm{NP} \subseteq \mathrm{DTIME}(T(n))$ or $\mathrm{NP} \subseteq \mathrm{RTIME}(T(n))$, with T polynomial or quasi-polynomial. These results are derived by reductions from interactive proofs. Namely, by first characterizing NP as those languages which have efficient interactive proofs of membership; and second by reducing the problem of whether there exists an interactive proof for membership in L ($L \in \mathrm{NP}$) to the problem of approximating the cost of an optimal solution to an instance of problem P.

Today such results are known for many important problems P, with values of Q and T which differ from problem to problem; for example, it was shown by [LY1] that approximating the size of the minimum set cover to within $\Theta(\log N)$ implies $\mathrm{NP} \subseteq \mathrm{DTIME}(n^{\mathrm{polylog}\, n})$, and it was shown by [FGLSS, AS, ALMSS] that for some constant $c > 0$ approximating the size of a maximum clique in a graph within factor n^c implies that $\mathrm{P} = \mathrm{NP}$. The values of Q and T achieved depend on the efficiency parameters of the underlying proof system used in the reduction to optimization problem P. The precise manner in which Q and T depend on these parameters depends on the particular problem P and the particular reduction used.

	$r = r(n)$	$p = p(n)$	$a = a(n)$	$\epsilon = \epsilon(n)$	How (in a word)
(1)	$O(\log n)$	2	$O(1)$	$\frac{1}{2} + \epsilon_*$	[ALMSS]+[FRS]+[Fe]
(2)	$O(\log n)$	$O(k(n))$	$O(1)$	$2^{-k(n)}$	$O(k(n))$ [CW, IZ]-style repetitions of (1).
(3)	$O(k(n) \log^2 n)$	2	$O(k(n) \log^2 n)$	$2^{-k(n)}$	[FL]
(4)	$O(k(n) \log n) + \text{poly}(k(n), \log\log n)$	4	$\text{poly}(k(n), \log\log n)$	$2^{-k(n)}$	This paper.

Figure 1: **Number of random bits (r), number of provers (p), answer size (a) and error probability (ϵ) in results of the form** $\mathrm{NP} \subseteq \mathrm{MIP}_1[r, p, a, q, \epsilon]$. Here $k(n)$ is any function bounded above by $O(\log n)$ and ϵ_* is any positive constant.

The goal of this paper is to improve the values of Q and T in such reductions. Thus we need to reduce the complexity of the underlying proof systems. Let us begin by seeing what are the proof systems in question and what within these proof systems are the complexity parameters we need to consider.

1.1 PCP and MIP

Several variants of the interactive proof model have been used to derive intractability of approximation results. We focus on two of them. The first is the (single round version of the) multi-prover model of Ben-Or, Goldwasser, Kilian and Wigderson [BGKW]. The second is the "oracle" model of Fortnow, Rompel and Sipser [FRS], renamed "probabilistically checkable proofs" by Arora and Safra [AS]. In each case, we may distinguish five parameters which we denote by r, p, a, q and ϵ (all are in general functions of the input length n). In a multi prover proof these are, respectively, the number of random bits used by the verifier, the number of provers, the size of each prover's answer, the size of each of the verifier's questions to the provers, and the error probability. Correspondingly in a probabilistically checkable proof these are the number of random bits used by the verifier, the number of queries to the oracle, the size of each of the oracle's answers, the size of each individual query, and the error probability. We denote by $\mathrm{MIP}_1[r, p, a, q, \epsilon]$ and $\mathrm{PCP}[r, p, a, q, \epsilon]$ the corresponding classes of languages.

Note that the total number of bits returned by the provers or oracle is pa; we will sometimes denote this quantity by t. In some applications the important parameter is the expected value of t. To capture this we define PCP' as PCP except that the p parameter is the *expected* number of queries made by the verifier.

The parameters which are important in applications to approximation seem to be r, p, a, ϵ. However, as q is important in transformations of one type of proof system to another, we included it in the list.

We emphasize that the model (PCP or MIP_1) is not as important, in this context, as the values of the parameters p, r, a, ϵ. Although the parameterized versions of PCP and MIP_1 are not known to have equal language recognition power for a given r, p, a, ϵ,[1] most known reductions to approximation problems in one model are easily modified to work in the other as long as p, r, a, ϵ remain the same. Accordingly, we sometimes state results in terms of MIP_1 and sometimes in terms of PCP, and leave the translations to the reader.

We note that there may be a motivation to move to the PCP model when proving an approximation hardness result if one could prove results of the form $\mathrm{NP} \subseteq \mathrm{PCP}[r, p, a, q, \epsilon]$ which attain better values of the parameters than results of the form $\mathrm{NP} \subseteq \mathrm{MIP}_1[r, p, a, q, \epsilon]$.

1.2 New Proof Systems for NP

Our main result is the construction of low complexity, low error proof systems for NP which have only a *constant* number of provers. In its most general form the result is the following.

Theorem 1.1 *Let* $k(n) \leq O(\log n)$ *be any function. Then* $\mathrm{NP} \subseteq \mathrm{MIP}_1[r, 4, a, q, 2^{-k(n)}]$, *where* $r = O(k(n) \log n) + \text{poly}(k(n), \log\log n)$, $a = \text{poly}(k(n), \log\log n)$, *and* $q = O(r)$.

The table of Figure 1 summarizes previous results of the form $\mathrm{NP} \subseteq \mathrm{MIP}_1[r, p, a, q, \epsilon]$ in comparison with ours. We omit the question size q from the table because it is in all cases $O(r)$ and doesn't matter in reductions anyway. The main result of [ALMSS], which states that there is a constant t such that $\mathrm{NP} = \mathrm{PCP}[O(\log n), t, 1, O(\log n), 1/2]$, is incorporated as the

[1] It is easy to see that $\mathrm{MIP}_1[r, p, a, q, \epsilon] \subseteq \mathrm{PCP}[r, p, a, q, \epsilon]$, but the converse containment is not known to be true. When complexity is ignored the models are of course the same [FRS, BFL]. For explanations and more information we refer the reader to §2.

special case $k(n) = 1$ of the result shown in (2). The result shown in (1) is obtained as follows. First apply the transformation of [FRS] to the [ALMSS] result to get $\mathrm{NP} \subseteq \mathrm{MIP}_1[O(\log n), 2, t, O(\log n), 1 - 1/(2t)]$ where t is the constant from [ALMSS] as above. Then apply [Fe] to bring the error to any constant strictly greater than $1/2$, at constant factor cost in the other parameters.

Comparing these results with ours, we note the following features, all of which are important to our applications. First, by setting $k(n)$ to an appropriate constant, we can achieve *any* constant error using only logarithmic randomness and answer sizes, which improves the result shown in (1), where logarithmic randomness and answer size are only achievable for a particular constant error. Second, the number of provers we use is a constant independent of the error, which is not true of the result shown in (2). Finally, for small $k(n)$ and given error $2^{-k(n)}$ our randomness and answer sizes are smaller than those of the result of [FL] shown in (3); in particular, for $k(n) = \log^{o(1)} n$ we have $O(k(n) \log n)$ randomness and answer sizes, and for $k(n) = O(\log \log n)$ we have polyloglog(n) answer sizes. On the other hand, the number of provers we use is four rather than the two achieved in the results (1) and (3), but this will suffice for our applications to approximation.

The (constant) number of bits t that one needs to check in the [ALMSS] result $\mathrm{NP} \subseteq \mathrm{PCP}[O(\log n), t, 1, O(\log n), 1/2]$ is of the order of 10^4. Some reductions in this value were obtained by [PS]. Our improved complexity of four prover proofs for NP in terms of randomness and answer size for a given error, together with a careful analysis of the [ALMSS] construction and proofs enable us to obtain substantially smaller values than previously known. Specifically, focusing on the expected value, we show the following.

Theorem 1.2 $\mathrm{NP} = \mathrm{PCP}'[O(\log n), 29, 1, O(\log n), 1/2]$.

Both of the above results have applications to approximation.

1.3 Applications to Approximation

A "hardness of approximation" result for a particular optimization problem P is, as we said above, a proof that the existence of a polynomial time, factor Q approximation algorithm for P would imply something like $\mathrm{NP} \subseteq \mathrm{DTIME}(T(n))$ or $\mathrm{NP} \subseteq \mathrm{RTIME}(T(n))$. Our new proof systems lead to improvements in the values of the quality Q of the approximation shown to be hard, and the time T for the deterministic or probabilistic simulation of NP in the conclusion, in known results for a variety of problems P. Specifically, we have improvements in both factor Q and conclusion T for set cover and quartic programming (using Theorem 1.1), and we have improvements in the factor Q for maximum clique, chromatic number, MAX3SAT, and quadratic programming (using Theorem 1.2).

Set Cover. For a definition of the problem we refer the reader to §4.1. Recall that there exists a polynomial time algorithm for approximating the size of the minimum set cover to within a factor of $\Theta(\log N)$, where N is the number of elements in the base set [Jo, Lo]. Hardness of approximation was recently shown by Lund and Yannakakis [LY1]. The reduction they used has the property that T is proportional to 2^{r+a} while Q increases as ϵ decreases, and decreases as $r + a$ increases.[2] Ideally we would like $r + a$ to be $O(\log n)$ (so that the conclusion is that approximation to within Q is NP-complete) while making ϵ as small as possible (so that Q can tend to its optimal value of $\Theta(\log N)$). However, the reduction, like those of [BR, FL, Be], additionally requires that the number of provers be $p = 2$, although (in this case) it can be extended easily to any constant. Lund and Yannakakis were thus constrained to use either the result $\mathrm{NP} \subseteq \mathrm{MIP}_1[O(\log n), 2, O(1), O(\log n), \theta(1)]$, or the result $\mathrm{NP} \subseteq \mathrm{MIP}_1[O(k(n)\log^2 n), 2, O(k(n)\log^2 n), O(k(n)\log^2 n), 2^{-k(n)}]$ (cf. (1) and (3) of Figure 1). Using the first they conclude that there exists some (particular) constant $c > 0$ such that approximating the value of the minimum set cover to within factor c is NP-complete. Using the second they conclude that approximating the value of the minimum set cover to within $\Theta(\log N)$ implies $\mathrm{NP} \subseteq \mathrm{DTIME}(n^{\mathrm{polylog}\, n})$. Based on Theorem 1.1 we can improve these results, as follows.

Theorem 1.3 *Let $c > 0$ be any constant. Suppose there is a polynomial time algorithm which approximates the size of the minimum set cover to within c. Then* $\mathrm{P} = \mathrm{NP}$.

Theorem 1.4 *For any constant $c < 1/8$ the following is true. Suppose there is a polynomial time algorithm which approximates the size of the minimum set cover to within $c \log N$. Then* $\mathrm{NP} \subseteq \mathrm{DTIME}(n^{\log\log n})$.

Similar improvements follow for all of the following problems: dominating set, hitting set, hypergraph transversal, minimum exact cover (cf. [KT, LY1]).

Max Clique, Chromatic number. Known results for Max-Clique are based on the result of [ALMSS] which states that there are constants t, d such that $\mathrm{NP} = \mathrm{PCP}[d \log n, t, 1, O(\log n), 1/2]$. Using the reduction of [FGLSS], it follows that there exists a constant

[2] This is a simplification of the actual situation, but will suffice for the purpose of this discussion. See Lemma 4.1 for the precise tradeoff.

$c > 0$ such that approximating the size of the maximum clique in a graph to within n^c is NP-complete; this result uses randomness efficient error reduction techniques such as [CW, IZ, BGG], and c depends on t, d as well as other constants arising from the error-reduction. Zuckerman [Zu] uses a random construction which achieves $c = 1/(1+t)$ at the cost of weakening the conclusion to NP $\subseteq$ BPP. Based on Theorem 1.2, we can improve the value of c in this result.

Theorem 1.5 *Suppose there exists a polynomial time algorithm which can approximate the size of the maximum clique in a graph to within* $n^{1/30}$. *Then* NP $\subseteq$ BPP.

The factor can be increased to $n^{1/25}$ if the conclusion is weakened to NEXP $\subseteq$ BPEXP. For the problem of approximating the chromatic number of a graph, the reduction of [LY1] implies the following.

Theorem 1.6 *Suppose there exists a polynomial time algorithm which can approximate the chromatic number of a graph to within* $n^{1/146}$. *Then* NP $\subseteq$ BPP.

The factor can be increased to $n^{1/121}$ if the conclusion is weakened to NEXP $\subseteq$ BPEXP.

MAX-3SAT. Using the same characterization of NP as for Max-Clique, it was also shown by [ALMSS] that there exists a constant $c > 1$ such that approximating the maximum number of simultaneously satisfiable clauses in a 3SAT instance to within factor c is NP-complete. We can prove the same with a higher value of c than previously achieved.

Theorem 1.7 *Suppose there exists a polynomial time algorithm which approximates the maximum number of simultaneously satisfiable clauses in a 3SAT instance to within* 113/112. *Then* P = NP.

The factor can be improved to 94/93 if the conclusion is weakened to EXP = NEXP.

QUARTIC PROGRAMMING. This is the problem of maximizing a n-variate polynomial of total degree four over a (compact) subset of R^n specified by linear constraints. We know that there exists a constant $c \in (0,1)$ such that the following is true: the existence of a polynomial time algorithm for c-approximating[3] the maximum of a quartic program implies P = NP; this follows from the corresponding result of [BR, FL] for quadratic programming. Based on Theorem 1.1 we can improve this to show the same for *any* constant.

[3] The definition of approximation used in this context of "continuous" optimization problems is not the usual one of approximating to within a multiplicative factor; see §4.3 for details.

Theorem 1.8 *Let* $c \in (0,1)$ *be any constant. Suppose there exists a polynomial time algorithm for* c*-approximating the maximum of a quartic program. Then* P = NP.

1.4 Open Problems

We believe that the optimal result to reach is NP $\subseteq$ $\mathrm{MIP}_1[O(\log n), 2, O(\log n), q(n), 1/n]$ for some $q(n)$. That is, NP has two prover proof systems with logarithmic randomness and answer sizes, and $1/n$ error. One implication would be that $(1 - n^{-\Theta(1)})$-approximating a quadratic program implies P = NP. For set cover, a weaker result would suffice. If for some constant p and $\epsilon(n) = 1/\log^{\omega(1)} n$ we could show that NP $\subseteq$ $\mathrm{MIP}_1[O(\log n), p, O(\log n), q(n), \epsilon(n)]$ for some $q(n)$, then we would get: approximating the size of a minimum set cover to within $c \log N$ implies P = NP for any constant $c < 1/(2p)$.

1.5 Notes

This is essentially the STOC version of our paper but corrects a few mistakes which have been pointed out to us subsequently. The changes are summarized below.

In the STOC version we said that the variant of the [FGLSS] clique reduction given in [Zu] shows that approximating clique to within $n^{1/t}$ implies NP $\subseteq$ BPP, and thus had Theorem 1.5 with a factor of $n^{1/29}$. In truth [Zu] achieves $n^{1/(1+t)}$ so that the factor in Theorem 1.5 should be the $n^{1/30}$ which now appears there. Thanks to David Zuckerman for pointing this out.

In the STOC version we implied that our result for clique implied a result for chromatic number with the same approximation factor. We had forgotten to factor in the cost of the reduction of [LY1] which increases the size of the graph. So the factor for chromatic number is $n^{1/146}$ as now stated in Theorem 1.6. Thanks to Madhu Sudan for pointing this out.

In the STOC version we said that the best known value of the constant error ϵ achievable for the result NP $\subseteq$ $\mathrm{MIP}_1[O(\log n), 2, O(1), O(\log n), \epsilon]$ was a value close to 1, namely $\epsilon = 1 - 1/(2t)$ where t is the number of bits queried in the [ALMSS] proof. However the result of [Fe] can be used to bring ϵ to any constant greater than 1/2, as now indicated in Figure 1(1). Thanks to Uri Feige for pointing this out.

2 Preliminaries

We extend notations and terminology for multi-prover and probabilistically checkable proofs to take into account the five measures of complexity (r, p, a, q, ϵ) in which we will be interested.

In a (one round) multi-prover proof system for a language L, a verifier V interacts on common input $x \in \{0,1\}^n$ with $p = p(n)$ provers. The interaction consists of a single round. Namely, V flips $r = r(n)$ coins to get a random string R, and as a function of x and R computes p queries $Q_1, \ldots, Q_p$, each of length $q = q(n)$. Q_i is sent to the i-th prover, who is regarded as a function from $\{0,1\}^q$ to $\{0,1\}^a$. This prover responds with a string A_i of length $a = a(n)$. As a function of $x, R, A_1, \ldots, A_p$ the verifier decides whether or not to accept. Usually the common input x and its length n will be understood. We will denote by $Q_V(R,i)$ the question posed to the i-th prover when the verifier's coins are R; this is computable in time polynomial in n. We let $Q_V(R) = (Q_V(R,1), \ldots, Q_V(R,p))$. We denote by $V(R, A_1 \ldots A_p)$ the verifier's decision on whether or not to accept. We demand that this decision procedure can be expressed as a circuit $C_R(A_1 \ldots A_p)$ of size poly(p, a), where the verifier accepts if and only if $C_R(A_1 \ldots A_p) = 1$. We require that C_R be constructible from x, R in time poly(n). We say that V is a $\mathrm{MIP}_1[r, p, a, q, \epsilon]$ verifier for L if (1) when $x \in L$ there are provers who can make the verifier accept with probability 1, while (2) if $x \notin L$ then for all collections of provers, the probability that the verifier accepts is at most the specified error-probability $\epsilon = \epsilon(n) < 1$. $\mathrm{MIP}_1[r, p, a, q, \epsilon]$ denotes the class of languages possessing $\mathrm{MIP}_1[r, p, a, q, \epsilon]$ verifiers. We let $\mathrm{MIP}_1 = \mathrm{MIP}_1[\mathrm{poly}(n), \mathrm{poly}(n), \mathrm{poly}(n), \mathrm{poly}(n), 1/2]$. It is known that $\mathrm{MIP}_1 = \mathrm{NEXP}$ [BFL].

Probabilistically checkable proofs, as defined in [AS, ALMSS] are the same as what [FRS] had earlier called the oracle model. In this model, a verifier V has access to an input $x \in \{0,1\}^n$ and an oracle which is regarded as a function from $\{0,1\}^q$ to $\{0,1\}^a$. The verifier flips r coins, as a function of which he computes p queries $Q_1, \ldots, Q_p \in \{0,1\}^q$ for the oracle. The oracle returns answers $A_1, \ldots, A_p \in \{0,1\}^a$, and the verifier decides whether or not to accepts as a function of $x, R, A_1, \ldots, A_p$. Attributes of the verifier are as for the multi-prover case. V is said to be a $\mathrm{PCP}[r, p, a, q, \epsilon]$ verifier for L if (1) when $x \in L$ there is an oracle which makes the verifier accept with probability 1, while (2) if $x \notin L$ then for any oracle the probability that the verifier accepts is at most the specified error-probability $\epsilon = \epsilon(n)$. Conventions and notation used will also be analogous to the MIP_1 case. In particular, $\mathrm{PCP}[r, p, a, q, \epsilon]$ denotes the class of languages possessing $\mathrm{PCP}[r, p, a, q, \epsilon]$ verifiers, and $\mathrm{PCP} = \mathrm{PCP}[\mathrm{poly}(n), \mathrm{poly}(n), \mathrm{poly}(n), \mathrm{poly}(n), 1/2]$. We also let $\mathrm{PCP}[r, p] = \mathrm{PCP}[r, p, 1, O(r), 1/2]$ as defined in [AS].

It is known that $\mathrm{PCP} = \mathrm{MIP}_1$ [FRS]. Furthermore, it is easy to see that $\mathrm{MIP}_1[r, p, a, q, \epsilon] \subseteq \mathrm{PCP}[r, p, a, q, \epsilon]$. But whether or not the converse containment is true remains an open question. In particular, known simulations of probabilistically checkable proofs by multi-prover ones (such as those used by [FRS] to show $\mathrm{PCP} = \mathrm{MIP}_1$) don't preserve complexity.[4]

Let us now state the (well known) lemma which will be our starting point. It is obtained by applying standard transformations (cf. [BGKW, FRS]) to the $\mathrm{NP} \subseteq \mathrm{PCP}[O(\log n), O(1), 1, O(\log n), 1/2]$ result of [ALMSS].

Lemma 2.1 *Suppose* $k(n) \leq O(\log n)$. *Then* $\mathrm{NP} \subseteq \mathrm{MIP}_1[O(k(n) \log n), O(k(n)), O(1), O(\log n), 2^{-k(n)}]$.

Note we could further reduce the randomness to $O(\log n)$ by using the techniques of [CW, IZ]; this is the result we stated in Figure 1(2). But the advantage will be lost in the transformations we will apply later, so we don't bother.

3 Efficient Proof Systems

We prove our main theorem in two steps. The starting point is the proof system of Lemma 2.1. This proof system has $O(k(n))$ provers, and we need to reduce this to a constant. Applying the transformation of [LS, FL] will not suffice, because in reducing the number of provers this transformation increases the amount of randomness as well as the answer sizes (cf. Figure 1(3)). We tackle this problem by concentrating first on keeping the randomness down. In a first step we show how to reduce the number of provers to two while not increasing the randomness, but at the cost of an *exponential* blowup in answer sizes. However, in a second step we apply recursion to reduce the answer sizes while keeping the other quantities under control. Let us now proceed to our first step.

3.1 Reducing Randomness

As discussed above, the transformation of [FL, LS] reduces the error without increasing the number of provers, but costs in randomness. We combine this transformation with the idea of [BFLS] of using as "base field" not $\{0,1\}$ but some larger subset H of the underlying finite field F. Choosing $h = \log |H|$ appropriately we need use only $O(k(n) \log n)$ random bits to get error $2^{-k(n)}$ at the cost of answer sizes which grow exponentially in h.

[4] The basic result of [FRS] is $\mathrm{PCP}[r, p, a, q, \epsilon] \subseteq \mathrm{MIP}_1[r + \lg p, 2, a, q, 1 - (1-\epsilon)/p]$.

Theorem 3.1 *Let $k = k(n) \leq O(\log n)$ be any function. Let $h(n) = \max(k(n), \log\log n)$. Then* $\mathrm{NP} \subseteq \mathrm{MIP}_1[k(n)\log n, 2, k(n)\log n \cdot 2^{h(n)}, k(n)\log n, 2^{-k(n)}]$.

Proof: Let $L \in \mathrm{NP}$. By Lemma 2.1 there is a $\mathrm{MIP}_1[r,p,a,q,\epsilon]$ verifier V for L with $p = O(k(n))$, $r = O(k(n)\log n)$, $q = O(\log n)$, $a = O(1)$, and $\epsilon = 2^{-k(n)-1}$.

We construct a $\mathrm{MIP}_1[r', 2, a', q', 2\epsilon]$ verifier V' for L, with r', q', a' to be specified later, as follows. Let $s = q/h$, $d = 2^h - 1$, and $H = \{0,1\}^h$. Fix a finite field $F \supseteq H$ with $f \overset{\text{def}}{=} \log|F| \overset{\text{def}}{=} 3h + \log(q/h)$. V' flips coins to get a r bit random string R, and for $i = 1, \ldots, p$ lets $Q_i = Q_V(R, i) \in \{0,1\}^q$ be the query that V would send to prover i. Regard each Q_i as divided into s pieces, each of size h (that is, $Q_i = Q_i^1 \ldots Q_i^s$ with each Q_i^j in H) so that Q_i is an element of $H^s \subseteq F^s$. We now apply the technique of [LS]. V' chooses, randomly and uniformly, $y_1, \ldots, y_p$ from F^s. For each i we let $l_i \colon F \to F^s$ be (a canonical representation of) the unique line through Q_i and y_i. Let $t_{i,1}, t_{i,2} \in F$ satisfy $l_i(t_{i,1}) = Q_i$ and $l_i(t_{i,2}) = y_i$. V' sends $l_1, \ldots, l_p$ to the first prover, and $y_1, \ldots, y_p$ to the second.

It is assumed that the first prover's reply has the form $(A_1, \ldots, A_p)$ where each $A_i \colon F \to F^a$ is a degree sd polynomial, and the second prover's reply has the form $(\alpha_1, \ldots, \alpha_p)$ with each α_i in F^a. For $i = 1, \ldots, p$ the verifier V' sets $a_i = A_i(t_{i,1})$. It also sets a bit CT ("consistency check") to 1 if for all $i = 1, \ldots, p$ it is the case that $A_i(t_{i,2}) = \alpha_i$, and 0 otherwise. V' accepts iff $CT = 1$ and $V(R, a_1 \ldots a_p) = 1$.

For each i let $g_i \colon H^s \to \{0,1\}^a$ represent an optimal strategy for the i-th prover in the protocol of V. Let $\hat{g}_i \colon F^s \to F^a$ denote an extension of g_i to a multinomial of degree $d = 2^h - 1$ in each variable. Suppose the provers are honest. Then the first prover chooses his reply polynomials according to $A_i = \hat{g}_i l_i$ for $i = 1, \ldots, p$ and the second prover chooses his reply points according to $\alpha_i = \hat{g}_i(y_i)$ for $i = 1, \ldots, p$. It is easy to see that these answers lead V' to accept with probability 1.

The amount of randomness used is $O(k(n)\log n)$ to generate R, plus $psf = O(k) \cdot (q/h) \cdot [3h + \log(q/h)]$ to generate $y_1, \ldots, y_p$. The latter is $\leq O(kq) + O(k) \cdot \log q/h$, which is $O(k(n)\log n)$ (because $h(n) \geq \log\log n$ and $q = O(\log n)$), so total randomness used is $r' = O(k(n)\log n)$ as desired. Query size q' is clearly of the same order. The answer size a' is dominated by the answer size for the first prover, which is $p(sd)(af) = O(k) \cdot (q2^h/h) \cdot [3h + \log(q/h)]$. Since $h(n) \geq \log\log n$ and $q = O(\log n)$ this is $O(k(n)\log n \cdot 2^{h(n)})$ as desired.

To prove the soundness, we need to trace through the proof of [LS], making the appropriate modifications. The error stemming from the construction of [LS] (i.e. the probability that the provers' replies are not "functional") can be bounded by

$$p\sqrt{\frac{sd}{2^f}}$$

which is at most $2^{-h-1} \leq 2^{-k-1} = \epsilon$ for large enough n. So the total error is at most $2\epsilon = 2^{-k}$. We omit the details. ▮

3.2 Reducing Answer Sizes

Answer sizes will be reduced by recursion (cf. [AS]). First need to define carefully how we look at MIP_1 verifiers. Let V be a $\mathrm{MIP}_1[r,p,a,q,2^{-k(n)}]$ verifier for some language L. Note that any strategy of a prover in the single question model is a (deterministic) function from questions to answers. We view the verification process as a family of circuits $\{C_R(y_1, \ldots, y_p)\}$ each of size $\mathrm{poly}(p, a)$; one for every random seed of V. The variables y_i will correspond to the ith prover as follows: any assignment Y_i to y_i will correspond to a strategy of the ith prover (i.e specifying all answers to all question). For every query Q, that V can ask the ith prover, the input variable y_i to C_R contains a segment of variables $(y_{iQ}^1, \ldots, y_{iQ}^a)$ such that y_{iQ}^j corresponds to the jth bit of the ith provers answer to the query Q. We say that the input y_i is *a-segmented*, since the length of each segment is bounded by a. Note that C_R for every R only depends on one segment from each input y_i. A family of circuit C_R with this property we denote by $\{C_R([y_1], \ldots, [y_p])\}$, i.e., there is a way of segmenting the inputs y_i such that $C_R(y_1, \ldots, y_p)$ only depend on one segment from y_i for any R.

First we will describe an encoding scheme of n bit strings. Let $m = \lceil \log n / \log\log n \rceil$ and F be a finite field such that $|F| \geq \log n$. Let $H = \{1, 2, \ldots \lceil \log n \rceil\} \subset F$. Since $|H^m| \geq n$ we can use H^m as indexes of the bits, i.e., we fix an injective map $\tau : \{1, 2, \ldots, n\} \to H^m$. Note that for any n bit string Y there exists a multivariate polynomial $\tilde{Y}(x_1, x_2, \ldots, x_m)$ such that the degree of any variable is at most $d = |H| - 1$ and such that for every $i \in \{1, 2, \ldots, n\} : Y_i = \tilde{Y}(\tau(i))$. We call $E_P(Y) = \langle \tilde{Y}(\alpha_1, \ldots, \alpha_m) \rangle_{(\alpha_1, \ldots, \alpha_m) \in F^m}$ the low degree encoding of Y. Given any two functions $f, f' : F^m \to F$ we define $\Delta(f, f') = |\{u \in F^m | f(u) \neq f'(u)\}| / |F|^m$, i.e., the fraction of points where f and f' disagree.

The following lemma describes that a given circuit C with p n-bit inputs it is possible to construct a family of smaller circuits that verifies in a probabilistic sense that C outputs 1. The circuit family corresponds to a $p+2$ prover MIP_1-proof system, such that if the first p provers answer accordingly to the low degree encoding of assignments $X_1, \ldots, X_p$ such that $C(X_1, \ldots, X_p) = 1$

then there exist strategies for the two last provers such that the MIP$_1$-verifier always accepts. On the other hand if the first p provers' answers are not close to a low degree encoding of some satisfying assignment to C, then for all possible strategies of the last two provers the MIP$_1$-verifier rejects with very high probability.

Lemma 3.2 (Circuit Verification) *Given a circuit $C(x_1, x_2, \ldots, x_p)$ and $k'(n)$ then there exists a polynomial-time computable (from C and $k'(n)$) circuit family*

$$\{C'_{R'}([x'_1], [x'_2], \ldots, [x'_p], [z_1], [z_2])\}$$

with such that $|C'_{R'}| = \text{poly}(\log |C|, k'(n))$, the inputs are $\text{poly}(\log |C|, k'(n))$-segmented and, $|R'| = \text{poly}(\log |C|, k'(n))$. Furthermore:

(1) *$\forall$ assignments $X_1, \ldots, X_p$, if $C(X_1, \ldots, X_p) = 1$ then there exist assignments Z_1, Z_2 such that $C'_{R'}(E_P(X_1), \ldots, E_P(X_p), Z_1, Z_2) = 1\ \forall\ R'$.*

(2) *$\forall$ assignments $X'_1, \ldots, X'_p, Z_1, Z_2$, if $\forall$ assignments $X_1, X_2, \ldots, X_p$ such that $C(X_1, \ldots, X_p) = 1$ there exists an $1 \leq i \leq p$ such that $\Delta(X'_i, E_P(X_i)) \geq 1 - 2^{-k'(n)}$, then $Pr_{R'}[C'_{R'}(X'_1, \ldots, X'_p, Z_1, Z_2) = 1] \leq 2^{2-k'(n)}$.*

Proof: The proof combines ideas in [BFLS, BLR, FRS, LS, FL].

Step 1: First we obtain a circuit family

$$\{C''_{R''}([x'_1], [x'_2], \ldots, [x'_p], z)\},$$

where the circuit size and number of additional random bits are $\text{poly}(\log |C|, k'(n))$. The notation implies that the first p inputs can be segmented in such a way that $C''_{R''}$ only depends on one segment for each input. For the last input z this is not the case as we shall see.

The basic idea is that z "proves" that C would output 1 when the inputs to C are the x'_i inputs *decoded*. The "proof" contain bit strings $X_1, \ldots, X_p$ and strings $X''_1, \ldots, X''_p$ over the alphabet F for some finite field F. The string X_i should equal the decoding of an assignment X'_i to the variables x'_i. The string X''_i should equal $E_P(X_i)$. The proof z contains W' the concatenation W of all these strings encoded using the low-degree encoding. The other part of z contains a probabilistically checkable proof W'' that all the strings have the right properties (i.e., that there exists a $W = (X_1, \ldots, X'_p)$ such that $\Delta(W', E_P(W)) < 1/3$, $C(X_1, \ldots, X_p) = 1$ and for every i that $X''_i = E_P(X_i)$). We use the construction of [BFLS] to construct these proofs. From [BFLS] it follows that we verify with probability 1/2 that W' has the correct properties by using $\text{poly}(\log |C|)$ random bits, reading $\text{poly}(\log |C|)$ bits and performing a computation that corresponds to circuits of size $\text{poly}(\log |C|)$.

Lastly, assuming that there exists some W such that $\Delta(W', E_P(W)) < 1/3$, we use the idea of self-correction [BLR, GS], which allows us to probabilistically access the segments of X''_i. For any segment s of W using $\text{poly}(\log |W'|)$ random, reading $\text{poly}(\log |W'|)$ bits from W' we can compute W_s by a circuit of size $\text{poly}(\log |W'|)$ with probability 1/2.

The circuits $C''_{R''}$ will probabilistically verify that Z encodes $W = (X_1, \ldots, X_p, X''_1, \ldots, X''_p)$ that satisfies the properties. Furthermore they will probabilistically verify that for every i that a random segment of X'_i equals the same segment of X''_i. All the tests are performed so as that the error is at most $2^{-k'(n)}$. (We perform all the tests $O(k'(n)$ times.)

By the construction it follows that if $C(X_1, \ldots, X_p) = 1$ then there exist assignments Z_1, Z_2 such that $C'_{R'}(E_P(X_1), \ldots, E_P(X_p), Z_1, Z_2) = 1$ for all R'.

On the other hand if $\Pr_{R'}[C'_{R'}(X'_1, \ldots, X'_p, Z_1, Z_2) = 1] > 2^{-k'(n)}$ then it implies that there exists assignments $X_1, \ldots, X_p$ such that $C(X_1, \ldots, X_p) = 1$ and $\Delta(X'_i, E_P(X_i)) < 2^{-k'(n)}$ for every $i = 1, \ldots, p$.

Step 2: Next we transform the C'' circuits into circuits $\{C'''_{R'',R'''}([x'_1], \ldots, [x'_p], [z^{(1)}], \ldots, [z^{(p')}])\}$ (i.e., a one-round $p + p'$-prover proof system) as follows using the standard method in [FRS]. (For every $j = 1, 2, \ldots, p'$, $Z^{(j)}$ should equal Z.) Assume that $C''_{R''}$ depends on the variables $z_{i_1}, \ldots, z_{i_q}$. Note that $q \leq |C''_{R''}|$. $C'''_{R'',R'''}$ partitions randomly $\{1, 2, \ldots, p'\}$ into q subset $S_1, \ldots, S_q$. Then $C'''_{R'',R'''}$ reads $z^{(j)}_{i_l}$ for every $l = 1, \ldots, l$ and $j \in S_l$. If for some l and $j, j' \in S_l$, $z^{(j)}_{i_l} \neq z^{(j')}_{i_l}$ then $C'''_{R'',R'''}$ outputs 0. Otherwise it simulates $C''_{R''}$. It can now be shown that the increase in error probability is less than $2^{-k'(n)}$, when $p' = \text{poly}(k'(n), |C''_{R''}|)$.

Step 3: Lastly we transform the C''' circuits into the C' circuits using the same construction as in Theorem 3.1. I.e., we use the transformation from p' provers into 2 provers in [LS, FL] on the inputs $z^{(j)}$ to get segmented circuits with $p + 2$ inputs. This will use an additional $O(p' \log(|z|) k'(n))$ random bits. Using the same analysis as in the proof of Theorem 3.1 we get the circuits $C'_{R'}$, where the increase in error probability can be made less than $2^{-k'(n)+1}$. ▌

Theorem 3.3 $\text{MIP}_1[r, p, a, q, 2^{-k(n)}] \subseteq \text{MIP}_1[r + \text{poly}(\log a, k), p + 2, \text{poly}(\log a, k), r + \text{poly}(\log a, k), 2^{-(k(n)/p)+3}]$.

Proof: Let V be a $\mathrm{MIP}_1[r, p, a, q, 2^{-k(n)}]$ verifier for some language L. Let $\{C_R([y_1], \ldots, [y_p])\}$ be the corresponding circuit family. We will now construct a circuit family $\{C_{R,R'}([y'_1], \ldots, [y'_p], [z_1], [z_2])\}$, such that for every segment y_{is} of y_i there will be a *meta-segment* y'_{is} of y'_i. This meta-segment supposedly contains the encoding of an assignment to y_{is} and it is divided into segments; one for each point in F^m. The inputs z_j for $j = 1, 2$ are also divided into meta-segments z_{jR}; one for each R.

Using Lemma 3.2 with C equal to C_R we get a family of circuits $C'_{R'}(x'_1, \ldots, x'_p, x''_1, x''_2)$. Let $C_{R,R'}(y'_1, \ldots, y'_p, z_1, z_2)$ be the circuit obtained from $C'_{R'}$, where the variables x'_i are substituted by y'_{iQ_i}, where $Q_i = Q_V(R, i)$ and the variables x''_j are substituted by z_{jR}.

It follows by construction that if $x \in L$ then there exists assignments (i.e., provers) $Y'_1, \ldots, Y'_p, Z_1, Z_2$ such that $C_{R,R'}([Y'_1], \ldots, [Y'_p], [Z_1], [Z_2]) = 1$ for all R, R'.

On the other hand assume that $x \notin L$. Thus $\Pr_R[C_R(Y_1, \ldots, Y_p) = 1] \leq 2^{-k(n)}$ for all assignments $Y_1, \ldots, Y_p$.

First define for every meta-segment y'_{is} *good* decodings. This will be any string Y such that $\Delta(Y'_{is}, E_P(Y)) < 2^{-k'(n)}$. It follows from the Lemma 3.4 that for any segment there can not be more than $2^{k'(n)+1}$ such decodings, assuming that $|F|$ is large enough. Next for every $i = 1, \ldots, p$ define $l = 2^{k'(n)+1}$ assignments $Y_i^{(1)}, \ldots, Y_i^{(l)}$ such that for any segment s and any good decoding Y of Y'_{is} there exist a j such that of $Y = Y_{is}^{(j)}$.

For any R we say that R is good if there exist assignments $Y_1^{(j_1)}, \ldots, Y_p^{(j_p)}$ such that $C_R(Y_1^{(j_1)}, \ldots, Y_p^{(j_p)}) = 1$.

Note that if R is not good then from Lemma 3.2 $\Pr_{R'}[C_{R,R'}([Y'_1], \ldots, [Y'_p], [Z_1], [Z_2]) = 1) \geq 2^{-k'(n)+2}$. Thus $\Pr_{R,R'}[C_{R,R'}([Y'_1], \ldots, [Y'_p], [Z_1], [Z_2]) = 1)$ is at most $2^{-k'(n)+2} + 2^{(k'(n)+1)p} 2^{-k(n)} < 2^{-(k(n)/p)+3}$, when $k'(n) = \frac{k(n)+3}{p+1} - 1$ and $k(n) \geq 3p$. ▮

Lemma 3.4 *For any string Y' if $|F| > 2m(|H|-1)/\epsilon^2$ then $|\{Y | \Delta(Y', E_P(Y)) < \epsilon\}| < 2/\epsilon$.*

Theorem 1.1 now follows from combining Theorem 3.1 and Theorem 3.3.

Remark The precise values for the randomness and answer sizes in Theorem 1.1 resulting from this proof are $r(n) = O(k(n) \log n) + O(k(n)^2 \bar{k}(n))$ and $a(n) = O(k(n)^2 \bar{k}(n))$, where $\bar{k}(n) = \max(k(n), \log \log n)$.

3.3 An efficient PCP for SAT

In this subsection we construct a probabilistically checkable proof for SAT in which the expected number of bits read is 29. The proof will use Theorem 1.1 and ideas in [AS, ALMSS]. The main improvement over the construction of [ALMSS] is our improved recursion step (Lemma 3.2 and Lemmas 3.5 and 3.10) which allows us to combine proof systems with almost no increase in the error probability. Furthermore we have an improved analysis of the linearity test in [BLR] and of the matrix multiplication test for the special case needed in [ALMSS]. We will now sketch our construction. Some knowledge of the proof in [ALMSS] is assumed of the reader.

First we need some different encoding schemes. Let X be an n-bit string. The *robust encoding* $E_R(X)$ of X is an element of $\{0,1\}^{2^n}$, indexed by elements v of $\{0,1\}^n$, such that the v^{th} bit of $E_R(x)$ is $\sum_{i=1}^n v_i x_i \bmod 2$. Furthermore we will have another encoding scheme E that encodes n-bit strings into $f(n)$-bit strings such that $f(n)$ is a polynomial, $E(X)$ is polynomial-time computable and the following is true: if X, X' are distinct strings then $\Delta(E(X), E(X')) > 1/2 - \delta$ for some constant $\delta > 0$. Such encodings exist for any fixed $\delta > 0$.

Fix $\epsilon > 0$. From Theorem 1.1 we get for any input x a segmented circuit family $C_R([y_1], \ldots, [y_4])$ such that the size of the circuits are at most $\mathrm{poly}\log\log(n)$. Furthermore if $x \in$ SAT then there exist assignments $Y_1, \ldots, Y_4$ such that $C_R(Y_1, \ldots, Y_4) = 1$ for all R, otherwise $\Pr_R(C_R(Y_1, \ldots, Y_4) = 1) < \epsilon$ for all assignments.

We construct another circuit family $C'_{R,R'}(y'_1, \ldots, y'_4, z)$. Each of the circuits will have constant size. The inputs to the new circuits are divided into two groups. First there are four inputs $y'_1, \ldots, y'_4$ that correspond to the inputs to C_R. These inputs are meant to be $y_1 \ldots, y_4$ encoded such that each segment is encoded using the encoding scheme E.

The second group of inputs are divided into 2^r parts $\{z_R\}$, one for each R. The idea is that an assignment Z_R to z_R "proves" that C_R would output 1 when the inputs to C_R are the y'_i inputs decoded. More precisely Z_R contains 8 binary strings $A_1, \ldots, A_4$ and $B_1, \ldots, B_4$. The string A_i should equal to the segment s_i of y_i that C_R depend on. The string B_i should equal $E(A_i)$. Let D be a circuit with inputs $A_1, \ldots, B_4$ that computes the predicate "$C_R(A_1, \ldots, A_4) = 1$ and $B_i = E(A_i)$ for every $i = 1, \ldots, 4$." (Note that $|D| = \mathrm{poly}(|C_R|)$ since E is a polynomial-time computable encoding scheme.) Furthermore the "proof" contains one string W that contains a bit for every gate in D. This string can be seen as a witness for the facts that $D(A_1, \ldots, B_4) = 1$. Let $W' = (W_j W_k)_{j<k}$. If the proof is valid then Z_R the assignments to z_R equals $E_R(A_1, \ldots, B_4, W, W')$.

In [ALMSS] it was shown how to probabilistically verify such proofs. Assume that $(Y_1', \ldots, Y_4', \{Z_R\}_R)$ is such a proof and that $l = |A_1| + |A_2| + \cdots + |W'|$. Hence $|Z_R| = 2^l$ for every R. Let $0 < \delta, \delta' < 1/10$. For a fixed R and for $i = 1, \ldots, 4$ the circuits $C'_{R,R'}$ depends on Y'_{is_i}, where s_i is the segment of y_i that C_R depends on. Furthermore $C'_{R,R'}$ depends on Z_R. Let $\pi = Z_R$ for notational convenience. The verification has four parts, one for each the following properties:

Linearity Property (P_1^R): There exists $(A_1, \ldots, B_4, W, W')$ such that $\Delta(\pi, E_R(A_1, \ldots, B_4, W, W')) < \delta'$.

Quadratic Property (P_2^R): $W' = (W_j W_k)_{j<k}$ (assuming P_1^R).

Consistency Property (P_3^R): $D(A_1, \ldots, B_4) = 1$ (assuming P_1^R and P_2^R).

Input Properties (P_4^R): $\Delta(Y'_{is_i}, B_i) < 1/2 - \sqrt{\delta}$ for $i = 1, 2, 3, 4$ (assuming P_1^R, P_2^R and P_3^R).

The verification uses the following four tests:

Linearity Test (T_1^R): Pick uniformly at random $v, u \in_R \{0,1\}^l$ and check $\pi_u + \pi_v = \pi_{u+v}$.

Quadratic Test (T_2^R): Pick uniformly at random $s, t \in_R \{0,1\}^{|W|}$ and check $(s \cdot W)(t \cdot W) = (s@t \cdot W')$, where $s@t$ is the $|W'|$ bit string such that the (l,k)th bit is $s_l t_k + s_k t_l$. Since the values of W and W' is not directly accessible in π the technique of self-correction is used [BLR]. This technique give us a probabilistic method to access the bits encoded in π. The main result we use is the following: if $\Delta(\pi, E_R(X)) \leq \delta'$ then $\Pr_{u \in \{0,1\}^l}[\pi_{i-u} + \pi_u = X_i] \geq 1 - 2\delta'$. Thus the test is the following: Pick $v, u, w \in_R \{0,1\}^l$ and $s, t \in_R \{0,1\}^{|W|}$ and test that $(\pi_{s-u} + \pi_u)(\pi_{t-v} + \pi_v) = (\pi_{s@t-w} + \pi_w)$.

Consistency Test (T_3^R): In [ALMSS] they showed that from D and $s \in \{0,1\}^{|W|}$ it is possible to compute an index v_s of $E_R(A_1, \ldots, B_4, W, W')$ and a bit b_s such that for all W if $D(A_1, \ldots, B_4) = 0$ then $\Pr_s[E_R(A_1, \ldots, B_4, W, W')_{v_s} \neq b_s] = 1/2$. On the other hand if $D(A_1, \ldots, B_4) = 1$ then there exists W such that $\Pr_s[E_R(A_1, \ldots, B_4, W, W')_{v_s} = b_s] = 1$. Thus using self-correction the test is the following: Pick $s \in_R \{0,1\}^{|W|}$, $u \in_R \{0,1\}^l$ and test that $\pi_{v_s - u} + \pi_u = b_s$.

Input Tests (T_4^R): For every input segment s_i that C_R depends on pick uniformly at random $j \in \{1, \ldots, |Y'_{is_i}|\}$ and $u \in_R \{0,1\}^l$ and test that $Y'_{is_ij} = \pi_{B_{ij}-u} + \pi_u$.

The next lemma implies that when $x \notin L$ then we can assume that for almost all R that some property is false.

Lemma 3.5 *If $Pr_R[P_1^R, P_2^R, P_3^R$ and $P_4^R] > \epsilon/(2\sqrt{\delta})^4$ then $x \in L$.*

Proof: The following lemma implies that each segment of Y_i' can only be decoded to a constant number of different segments of Y_i.

Lemma 3.6 [MS] *Let $A = A(n, d, w)$ be the maximum number of binary vectors of length n, each with weight (i.e., number of non-zeroes) at most w, and any two of which are distance at least d apart. Then $A \leq dn/(dn - 2nw + 2w^2)$ if $dn - 2nw + 2w^2 > 0$.*

This implies for any Y'_{is_i} that $|\{B_i | \Delta(B_i, Y'_{is_i}) < 1/2 - \sqrt{\delta}\}| \leq 1/(2\sqrt{\delta})$. Thus, using the same arguments as in the proof of Theorem 3.3, the error probability is the old error probability times $1/(2\sqrt{\delta})^4$. ▮

The next lemma bounds the error probability of T_1^R:

Lemma 3.7 *If $\delta' = 1/4 - \sqrt{5}/12 = 0.06366$ and $\neg P_1^R$ then $Pr_{R'}[T_1^R$ rejects$] \geq 1/6$.*

Proof: The proof has two cases. The first case is that there does not exist some Y such that when $\Delta_{E_R}(\pi, E_R(Y)) < 1/3$ and the second case is when there exist some Y such that $\delta' < \Delta_{E_R}(\pi, E_R(Y)) < 1/3$. The first case follows from a Lemma in [BLR, ALMSS].

Hence assume that we are in case two. Let S be the set of $u \in \{0,1\}^l$ such that $\pi_u \neq E_R(Y)_u$. Let q be the probability that $\pi_u + \pi_v \neq \pi_{u+v}$ and let $s = |S|/2^l$. We want to show that $q \geq 3s - 6s^2$. Define the graph G as the complete directed graph with vertices $\{0,1\}^l$. We color the edges as follows: given an edge (u, v) we color it 1 if $\pi_u + \pi_v \neq \pi_{u+v}$ otherwise we color it with 0. Note that q is equal to the fraction of edges colored 1. Let G_S be the subgraph of G such that (u, v) is an edge in G_S if $v - u \in S$. Let $G_{\overline{S}}$ be the complement to G_S. The number of edge colored 1 in $G_{\overline{S}}$ is at least $2(|S||\overline{S}| - |S|^2)$, because the number of edge that leaves S and does not end in S is at least $|S||\overline{S}| - |S|^2$ (the number of edges from S minus the maximum number of edges from S to S). The number of edge colored 0 in G_S is at most $2(|S|^2)$, since there are at most $|S|^2$ edges from S to $\overline{S}$. Thus $q \geq (2(|S||\overline{S}| - |S|^2) + 2^l|S| - 2(|S|^2))/2^{2l} = 3s - 6s^2$. ▮

The following lemmas give bounds on the error probability for the last three tests. Their proofs are omitted in this abstract.

Lemma 3.8 *If P_1^R and $\neg P_2^R$ then the probability that T_2^R rejects is at least $\frac{3-\delta'^2}{8}(1 - 2\delta')$.*

Lemma 3.9 [ALMSS] *If P_1^R, P_2^R and $\neg P_3^R$ then the probability that T_3^R rejects is at least $(1/2)(1 - 2\delta')$.*

Lemma 3.10 *If P_1^R, P_2^R, P_3^R and $\neg P_4^R$ then the probability that T_4^R rejects is at least $(1/2 - \sqrt{\delta})(1 - 2\delta')$.*

Proof of Theorem 1.2: First let $\delta' = 0.06366$ and choose δ, ϵ small, say $\delta = 0.01$ and $\epsilon = 10^{-7}$.

Secondly, note that we do not need independence between the tests. This implies that we can reuse the random bit locations read in T_1^R in the other tests. This implies that all the self-correcting can be done reading no extra bits.

We will use the following protocol: Perform T_1^R 3 times with probability 0.18 otherwise 4 times (3 bits read each time), perform T_2^R once with probability 0.20 otherwise twice (3 bits each), T_3^R once with probability 0.70 otherwise twice (each times 1 bit) and T_4^R once with probability 0.70 otherwise twice (each time 8 bits). In all 28.56 bits on average.

We bound the error probability as follows: Because of the choice of how many tests to do we have that if for some R there exists some $j = 1, \ldots, 4$ such that P_j^R does not hold then the $C'_{R,R'}$ circuits will only output 1 with probability at most 0.4997. Lastly we know from Lemma 3.5 that if $x \notin L$ then the probability that the above will not happen is at most $\epsilon/(2\sqrt{\delta})^4 < 0.0003$. Thus the error probability is at most 1/2. ▌

4 Applications

Fix a combinatorial optimization problem and let $g(x)$ be the optimal value at input x. An approximation algorithm A is said to approximate $g(x)$ to within $f(N)$ if for every input x it is the case that

- $f(N)^{-1}g(x) \leq A(x) \leq g(x)$ if the problem is maximization
- $g(x) \leq A(x) \leq f(N)g(x)$ if the problem is minimization.

Here $N = \|x\| \in \mathsf{N}$ is the "norm" of the input x, i.e. some aspect of the input in terms of which approximation is measured. For example for set cover it is the number of elements in the base set.

We will now discuss some of the applications mentioned in §1.3. Specifically, we discuss set cover, max 3SAT, and the programming problems; the other applications are omitted due to lack of space.

4.1 Set Cover

We say that sets $S_1, \ldots, S_m$ cover a set U if $U \subseteq S_1 \cup \cdots \cup S_m$. An instance of the set cover problem consists of a base set U of size N and a sequence $S_1, \ldots, S_m$ of subsets of U. We denote by $\texttt{mincover}(\mathcal{S})$ the size of a minimum sized subcollection of $S_1, \ldots, S_m$ which covers the base set U.

Let V be a $\mathrm{MIP}_1[r, p, a, q, \epsilon]$ verifier for SAT. The set cover reduction (used to show that $\texttt{mincover}(\mathcal{S})$ is hard to approximate) requires that V satisfy some extra properties which we now detail. The first, *functionality*, requires that for each $R \in \{0,1\}^r$ and each $A_1 \in \{0,1\}^a$ there is at most one vector $(A_2, \ldots, A_p)$ with the property that $V(R, A_1 \ldots A_p) = 1$. The second, *uniformity*, requires that for each $i = 1, \ldots, p$ there is a set $\mathcal{Q}_i \subseteq \{0,1\}^q$ such that the queries of V to the i-th prover are uniformly distributed over $\mathcal{Q}_i$. The third, *equality of question space sizes*, requires that the sets $\mathcal{Q}_1, \ldots, \mathcal{Q}_p$ from the uniformity condition are all of the same size. The last, *disjointness of answer spaces*, requires that for each $i = 1, \ldots, p$ there is a set $\mathcal{A}_i \subseteq \{0,1\}^a$ such that $\mathcal{A}_1, \ldots, \mathcal{A}_p$ are disjoint and V rejects answers $A_1, \ldots, A_p$ from the provers if it is not the case that $A_1 \in \mathcal{A}_1, \ldots, A_p \in \mathcal{A}_p$. We call V *canonical* if all four properties hold.

We now state a general formulation of Lund and Yannakakis's reduction of a canonical proof system to set cover. In this lemma, l is a parameter to be chosen at will, and $\mathcal{Q}_i$ refers to the question spaces of the uniformity condition. The statement and proof of this lemma that appear in [LY1] are only for the case $p = 2$, but the authors say later that it extends to any constant, and this extension is indeed easy.[5] For completeness we provide the construction for this extension.

Lemma 4.1 *Suppose verifier V defines a canonical* $\mathrm{MIP}_1[r, p, a, q, \epsilon]$ *proof system for* SAT. *Fix a function $l\colon \mathsf{N} \to \mathsf{N}$. To any* SAT *instance φ of size n we can associate a set cover instance $\mathcal{S}_\varphi$ such that the following properties hold:*

(1) *If $\varphi \in$ SAT then* $\texttt{mincover}(\mathcal{S}_\varphi) \leq \sum_{i=1}^{p} |\mathcal{Q}_i|$.

(2) *If $\varphi \notin$ SAT then* $\texttt{mincover}(\mathcal{S}_\varphi) \geq (1 - \epsilon l^p) \cdot \frac{l}{p} \cdot \sum_{i=1}^{p} |\mathcal{Q}_i|$.

The size of the instance $\mathcal{S}_\varphi$ is $p2^{O(r+a+l)}$, and the transformation of φ to $\mathcal{S}_\varphi$ is computable in time polynomial in n and this quantity. The number of points in the base set of $\mathcal{S}_\varphi$ is $O(2^{r+2a+2l})$.

Proof: Let $\mathcal{A}_1, \ldots, \mathcal{A}_p$ denote the sets specified by the disjointness of answer spaces condition. For each $R \in \{0,1\}^r$ and each $A_1 \in \mathcal{A}_1$ let $\mathrm{UA}(R, A_1)$ denote the unique vector $(A_2, \ldots, A_p)$ such that $V(R, A_1 \ldots A_p) = 1$, if this vector exists (otherwise $\mathrm{UA}(R, A_1)$ is undefined). Let $m \stackrel{\text{def}}{=} 2^a \geq \sum_{i=1}^{p} |\mathcal{A}_i|$. Let $(B; B_1, \ldots, B_m)$ be a "set system" as per [LY1, Lemma 3.3]. The base set S of the set cover instance $\mathcal{S}_\varphi$ associated to φ is defined by $S = \{0,1\}^r \times B$. The sets in $\mathcal{S}_\varphi$ are the following.

[5] In fact, the extension does not even actually require that p be constant, and although it is constant in our applications, we thought it worthwhile to state the general lemma.

First, for each $Q \in \mathcal{Q}_1$ and each $A \in \mathcal{A}_1$ we have the set $S(1, Q, A)$ defined as

$$\{\langle R, b\rangle \in S : \text{UA}((R, A) = (A_2, \ldots, A_p) \text{ is defined}, \\ Q = Q_V(R, 1), \text{ and } b \in \overline{B}_{A_2} \cap \cdots \cap \overline{B}_{A_p}\} .$$

Second, for each $i = 2, \ldots, p$ and each $Q \in \mathcal{Q}_i$, $A \in \mathcal{A}_i$ we have the set $S(i, Q, A)$ defined as

$$\{ \langle R, b\rangle \in S : Q = Q_V(R, i) \text{ and } b \in B_A \} .$$

The proof that this construction works is a simple extension of the proof for the case $p = 2$ in [LY1]. ▌

We claim (proof omitted) that the verifier of Theorem 1.1 satisfies functionality and uniformity. Equality of question space sizes can then be achieved by a simple transformation as shown in [LY1], and disjointness of answer spaces by a simple padding; the cost is only a constant factor in randomness, question sizes, and answer sizes. Thus we have canonical $\text{MIP}_1[r, 4, a, q, 2^{-k(n)}]$ verifiers for SAT with r, q, a being as in Theorem 1.1. Based on this fact and Lemma 4.1 we can prove Theorems 1.3 and 1.4. Briefly, for the first, set $l = 2cp$ and choose $k(n) = O(1)$ to satisfy $2^{-k} l^p < 1/2$. For the second set $l = O(\log n \cdot \log \log n)$ and $k(n) = O(\log \log n)$.

4.2 Max-3SAT

We include a proof for Theorem 1.7 based on the construction in Section 3.3.

Proof of Theorem 1.7: Note that T_1^R needs 4 clauses, T_2^R needs 15 clauses, T_3^R needs 2 clauses and T_4^R needs 16 clauses. Now it follows that if we perform T_1^R with probability 1, T_2^R with probability 0.51, T_3^R with probability 0.385 and T_4^R with probability 0.385, then the overall error probability is at most 5/6. On average we have 18.52 clauses. This completes the proof since $18.58 \cdot 6 < 112$. ▌

4.3 Quartic Programming

We recall that quartic programming is the problem of maximizing a n-variate polynomial $f(x_1, \ldots, x_n)$ of total degree four, subject to linear constraints $Ax \leq b$. We will assume that the feasible region $\{ x \in \mathsf{R}^n : Ax \leq b \}$ is compact. Note that we are maximizing over a subset of R^n; solutions are not restricted to integers. We denote by f^* the maximum of f over the feasible region, and by f_* the minimum. Following [ADP, Va] we say that $\tilde{f}$ is a μ-approximation, where $\mu \in [0, 1]$, if $|f^* - \tilde{f}| \leq \mu \cdot |f^* - f_*|$. We refer the reader to [Va] for discussion of the appropriateness of this definition, and the inappropriateness of the one we have been using above, in the context of continuous optimization problems. Quartic programming is important in applications such as the "molecule embedding problem."

The reduction of two prover proofs to quadratic programming in [BR, FL] is easily extended to a reduction of four prover proofs to quartic programming. Theorem 1.1 guarantees for us four prover proof systems for NP with logarithmic randomness and answer sizes, and error an arbitrary constant. Put together these facts enable us to prove Theorem 1.8. We omit the details.

Acknowledgments

We are grateful to Dror Lapidot, for explaining to us the construction and proof of [LS]: these were crucial to the proof of Theorem 3.1. We thank Rajeev Motwani, Nick Reingold, Muli Safra, Madhu Sudan, Mario Szegedy, Steve Vavasis and Mihalis Yannakakis for their helpful insights.

Work done while the first author was at the IBM T.J. Watson Research Center, New York.

References

[AS] S. Arora and S. Safra. Approximating clique is NP-complete. *Proceedings of the 33rd Annual IEEE Symposium on the Foundations of Computer Science,* IEEE (1992).

[ALMSS] S. Arora, C. Lund, R. Motwani, M. Sudan and M. Szegedy. Proof verification and intractability of approximation problems. *Proceedings of the 33rd Annual IEEE Symposium on the Foundations of Computer Science,* IEEE (1992).

[ADP] G. Ausiello, A. D'Atri and M. Protasi. Structure preserving reductions among convex optimization problems. *Journal of Computer and System Sciences* **21**, 136–153 (1980).

[BFL] L. Babai, L. Fortnow and C. Lund. Non-Deterministic Exponential Time has Two-Prover Interactive Protocols. *Proceedings of the 31st Annual IEEE Symposium on the Foundations of Computer Science,* IEEE (1990).

[BFLS] L. Babai, L. Fortnow, L. Levin, and M. Szegedy. Checking Computations in Polylogarithmic Time. *Proceedings of the 23rd Annual ACM Symposium on the Theory of Computing,* ACM (1991).

[Be] M. Bellare. Interactive Proofs and Approximation: Reductions from Two Provers in One Round. *Proceedings of the 2nd Israel Symposium on Theory and Computing Systems*, IEEE (June 1993). Preliminary version: IBM Research Report RC 17969 (May 1992).

[BGG] M. Bellare, O. Goldreich and S. Goldwasser. Randomness in Interactive Proofs. *Proceedings of the 31st Annual IEEE Symposium on the Foundations of Computer Science,* IEEE (1990).

[BR] M. Bellare and P. Rogaway. The Complexity of Approximating a Nonlinear program. *Complexity of Numerical Optimization*, Ed. P.M. Pardalos, World Scientific (1993). Preliminary version: IBM Research Report RC 17831 (March 1992).

[BGKW] M. Ben-Or, S. Goldwasser, J. Kilian and A. Wigderson. Multi-Prover Interactive Proofs: How to Remove Intractability Assumptions. *Proceedings of the 20th Annual ACM Symposium on the Theory of Computing,* ACM (1988).

[BLR] M. Blum, M. Luby, and R. Rubinfeld. Self-testing and self-correcting programs, with applications to numerical programs. *Proceedings of the 22nd Annual ACM Symposium on the Theory of Computing,* ACM (1990).

[CW] A. Cohen and A. Wigderson. Dispersers, Deterministic Amplification, and Weak Random Sources. *Proceedings of the 30th Annual IEEE Symposium on the Foundations of Computer Science,* IEEE (1989).

[Co] A. Condon. The complexity of the max word problem, or the power of one-way interactive proof systems. STACS 91.

[FRS] L. Fortnow, J. Rompel and M. Sipser. On the power of multiprover interactive protocols. Structures 1988.

[Fe] U. Feige. On the Success Probability of the Two Provers in One Round Proof Systems. Structures 1991.

[FGLSS] U. Feige, S. Goldwasser, L. Lovász, S. Safra, and M. Szegedy. Approximating clique is almost NP-complete. *Proceedings of the 32nd Annual IEEE Symposium on the Foundations of Computer Science,* IEEE (1991).

[FL] U. Feige and L. Lovász. Two-Prover One Round Proof Systems: Their Power and their Problems. *Proceedings of the 24th Annual ACM Symposium on the Theory of Computing,* ACM (1992).

[GS] P. Gemmell and M. Sudan. Highly Resilient Correctors For Polynomials. *IPL*, 1992.

[IZ] R. Impagliazzo and D. Zuckerman. How to Recycle Random Bits. *Proceedings of the 30th Annual IEEE Symposium on the Foundations of Computer Science,* IEEE (1989).

[Jo] D. Johnson. Approximation algorithms for combinatorial problems. *J. of Computer and System Sciences* **9**, 256–278 (1974).

[KT] P. Kolaitis and M. Thakur. Approximation properties of NP minimization classes. *Structure in Complexity Theory*, 1991.

[LS] D. Lapidot and A. Shamir. Fully Parallelized Multi-Prover Protocols for NEXP-time. *Proceedings of the 32nd Annual IEEE Symposium on the Foundations of Computer Science,* IEEE (1991).

[Lo] L. Lovász. On the ratio of optimal integral and fractional covers. *Discrete Mathematics* **13**, 383-390 (1975).

[LY1] C. Lund and M. Yannakakis. On the Hardness of Approximating Minimization Problems. *Proceedings of the 25th Annual ACM Symposium on the Theory of Computing,* ACM (1993).

[LY2] C. Lund and M. Yannakakis. The Approximation of Maximum Subgraph Problems. ICALP 93.

[MS] F. MacWilliams and N. Sloane. *The Theory of Error-Correcting Codes.* North-Holland, 1981.

[PY] C. Papadimitriou and M. Yannakakis. Optimization, approximation, and complexity classes. *Proceedings of the 20th Annual ACM Symposium on the Theory of Computing,* ACM (1988).

[PS] S. Phillips and S. Safra. PCP and tighter bounds for approximating MAXSNP. Manuscript, 1992.

[Va] S. Vavasis. On approximation algorithms for concave programming. *Recent Advances in Global Optimization,* C. A. Floudas and P.M. Pardalos, pp. 3-18, Princeton University Press, 1992.

[Zu] D. Zuckerman. NP-Complete Problems have a version that is hard to Approximate. *Structure in Complexity Theory*, 1993.